

General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

AgRISTARS

SR-X1-04033
NAS9-15981

E83-10309

A Joint Program for
Agriculture and
Resources Inventory
Surveys Through
Aerospace
Remote Sensing

"Made available under NASA sponsorship
in the interest of early and wide dis-
semination of Earth Resources Survey
Program information and without liability
for any use made thereof."

Supporting Research

January 1981

ASSEMBLY LANGUAGE CODING FOR CLASSY

(E83-10309) ASSEMBLY LANGUAGE CODING FOR
CLASSY (Elogic, Inc.) 36 p HC A03/MF A01
CSCL 02C

N83-27299

Unclas
G3/43 00309

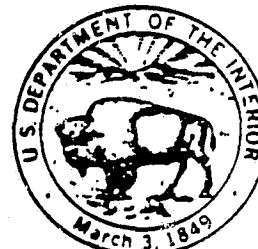
M. E. Rassbach



Elogic, Inc.
4242 S.W. Freeway, Suite 304
Houston, Texas 77027



NASA



Lyndon B. Johnson Space Center
Houston, Texas 77058

1. Report No. SR-X1-04033		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle Assembly Language Coding for CLASSY				5. Report Date	
				6. Performing Organization Code	
7. Author(s) M. E. Rassbach				8. Performing Organization Report No. NAS811	
9. Performing Organization Name and Address Elogic, Inc. 4242 S.W. Freeway, Suite 304 Houston, TX 77027				10. Work Unit No.	
				11. Contract or Grant No. NAS9-15981	
				13. Type of Report and Period Covered Technical Report	
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Lyndon B. Johnson Space Center Houston, TX 77058				14. Sponsoring Agency Code	
15. Supplementary Notes					
16. Abstract A set of assembly language improvements to the CLASSY clustering algorithm have been developed and tested. These are described in detail and analyzed.					
17. Key Words (Suggested by Author(s)) CLASSY algorithm			18. Distribution Statement		
19. Security Classif. (of this report) Unclassified		20. Security Classif. (of this page) Unclassified		21. No. of Pages 35	
				22. Price*	

*For sale by the National Technical Information Service, Springfield, Virginia 22161

SR-X1-04033
NAS9-15981

ASSEMBLY LANGUAGE CODING FOR CLASSY

BY

M. E. RASSBACH

This report describes Classification activities of the
Supporting Research project of the AgRISTARS program.

Elogic, inc.
4242 S.W. Freeway, Suite 304
Houston, Texas, 77027

January 20, 1981

CONTENTS

Purpose	1
Matrix Arithmetic	2
New Exponential Routine	5
Path Difference in CLASSY	9
Appendix 1: Program listings for vector routines .	11
Appendix 2: Program listings for the fast exponential system	26

Assembly language coding for CLASSY

Elogic has developed several assembly language routines to speed up the CLASSY algorithm.

These fall into two categories

- 1) matrix arithmetic
- 2) improvement in the exponential routine

Purpose of assembly-coded Matrix Arithmetic and Exponential Routines.

Currently available compilers produce a great number of extra instructions, particularly "overhead" instructions such as Load, Store, and index arithmetic. These instructions can be eliminated by careful hand coding of the routines. Thus it is possible to reduce the machine-processing time for a program by coding the most frequently used routines in assembly language.

CLASSY is particularly suited for this form of optimization, since most of its execution time is spent in a few vector and matrix processing routines. In addition, as was noted while designing the vector processing routines, application of the distributive law will allow putting one multiplication outside of the inner loop in several instances. (It was later realized that this change could be made in the Fortran version, as well.) Thus by using new, assembly language versions of the matrix arithmetic routines, CLASSY could be greatly accelerated. Some results are given in Table 1.

Matrix Arithmetic

The matrix arithmetic routines correspond exactly (bit-by-bit) to Fortran versions ("new Fortran") of the old matrix routines. The new Fortran versions differ from the original Fortran versions ("old Fortran") by the application of the distributive law and precalculation of some quantities to reduce the number of operations inside the inner loop. Rounding error differences can make the new version differ from the old version, which may become substantial due to the characteristics of the CLASSY algorithm (See "Path Differences in CLASSY" below). However, since these differences are only due to floating-point rounding error, which is random in both the new and old versions, the results are equivalent. The new version may in fact give slightly higher precision, due to the method of summation used. Table 2 gives the routine names for the new and old Fortran versions, as well as the assembly version. The names of EXEC'S used to switch from one version to another are also given. The names given are the names of the source language files; the object programs must have the original name.

<u>Version</u>	<u>Routines (Vector, XP)</u>	<u>Iterations, Channels</u>	<u>Time (Improve- ment)</u>	<u>Time per Cluster per Iteration (Improvement)</u>
Original	Old Fort., Fort. XP	10 4	369	4.10
Old Fort.	Old Fort., ALC XP	10 4	518 (0.71)	3.99 (1.03)
New Fort.	New Fort., ALC XP	10 4	301 (1.23)	3.34 (1.23)
ALC	ALC, ALC XP	10 4	240 (1.54)	2.67 (1.54)
Original	Old Fort., Fort XP	7 8	1590	15.14
Old Fort.	Old Fort., ALC XP	7 8	1164 (1.37)	18.47 (0.82)
New Fort.	New Fort., ALC XP	7 8	927 (1.72)	9.46 (1.60)
ALC	ALC, ALC XP	7 8	641 (2.48)	6.54 (2.31)

Table 1: Run time comparison.

This table shows a comparison of the run times of CLASSY using different combinations of routines. The run was 10 iterations on 4 channels of the simulated data, without any map generation. Initialization and randomization time has been subtracted off. Improvement is the speed ratio relative to the corresponding original version. (ALC means Assembly Language Coding, Fort. means Fortram.)

All runs were done on a simulated data set using a 4900 point data set. The scatter in the results is due to the effects of path differences (see heading on these).

Table 2: Routine names for matrix arithmetic.

<u>Routine</u>	<u>Old Fortran</u>	<u>New Fortran</u>	<u>Assembly</u>
DOTSQ	DOTSQB	DOTSQ	DOTSQA
CORECT	CORECTB	CORECT	CORECTA
MPVS	MPVSB	MPVS	MPVSA
VPV	VPV	VPV	VPVA
VMTV	VMTV	VMTV	VMTVA
EXEC to compile	VERSOLD	VERSNEW	VERSALC

New Exponential routine (XP)

In order to speed up executions, especially when the number of channels is few, Elogic has written a special exponential routine, called XP, specialized to the needs of the CLASSY algorithm. In CLASSY, the exponential is calculated as often as is the primary quadratic form, and the standard exponential routine requires several multiplies and a divide. The new exponential used for CLASSY reduces these times by table look up and by allowing a small increase in the RMS error. The new routine has an RMS precision of about 1 part in 24,000, which is easily adequate for the needs of CLASSY. In addition, the new routine uses linear interpolation and requires only one multiply.

The new exponential routine calculates $XP(x) = \exp(-x/2)$ by subdividing the interval between 1 and 16 into 240 equal parts. A table contains the coefficients of the least-squares fit of a linear form (First - order (polynomial) to the desired exponential in this region. These coefficients give a result accurate to about one part in 24,000 RMS in the result. Arguments outside the range of 1 to 16 are handled by directly obtaining the exponential of the remaining part and multiplying it times the part calculated above. Note that because of limitations on the exponent range of the result to the routine, arguments are limited to lie between $2 \log 10^{\pm 76}$. This routine will not raise an exponent underflow condition if the result is too small; rather, it returns

a result of zero. This saves additional time since STATIS will not be required to check the argument range before calculating the exponential.

The table used XP is currently calculated at execution time by the routine XPREP which must be called by CLINIT during initialization. (A version of CLINIT making this call must be used whenever the XP routine is to be used.) XP could be modified to contain these constants directly. A modified version must be also used of STATIS, which calls the XP routine.

The new exponential routine $XP(x) = \exp(-x/2)$ is thus a fast assembly language routine designed to calculate the oft-repeated exponential function quickly and with an accuracy easily sufficient for use in CLASSY.

The XP routine causes slight variations in the calculated probabilities in CLASSY, which can cause the algorithm to take a different path (See the note "Path Differences in CLASSY" below.)

XP routine

The exponential $e^{-x/2}$ is calculated as follows: x is divided into three parts, I , n , and Σ .

$$x = 16I + \frac{n}{16} + \Sigma$$

$$-22 \leq I \leq 21 \quad ; \quad 0 \leq n < 256 \quad ; \quad 0 \leq \Sigma < \frac{1}{16}$$

Then

$$\begin{aligned} e^{-x/2} &\approx \text{XPBIG}(I) * \text{XPAR}(n) * (\text{XPD} + \Sigma) \\ &\approx 0 \quad I < -22 \quad \text{or} \quad I > 21 \end{aligned}$$

This approximation has a relative RMS error of about $1/27000$, which is well adequate for the CLASSY statistical system. The variable OUTCNT counts the number of overflows made.

XPP, XPAR, and XPBIG are generated by the routine XPREP.

Equations for constants used in exponential routine:

$$\text{XPAR}(n) = d_1 e^{-2\alpha n - \alpha} \quad n = 0(1)256$$

$$\text{XPD} = d_0 / d_1 - 2\alpha$$

$$d_0 = \sinh \alpha / \alpha \quad \approx 1 + \frac{\alpha^2}{6} + \dots$$

$$d_1 = \frac{-3}{2\alpha^3} \quad (\sinh \alpha - \alpha \cosh \alpha) \approx -\frac{1}{2} - \frac{\alpha^2}{20} - \dots$$

$$\alpha = \frac{\beta}{4} = \frac{1}{64} \text{ for intervals of } \frac{1}{16} \text{ in original argument (of } e^{-x/2})$$

These coefficients are the result of a least-squares fit of $(d_0 + d_1 \Sigma)$ to $e^{-\Sigma/2}$ for $-\frac{\beta}{2} < \Sigma < \frac{\beta}{2}$

$$\text{XPBIG}(I) = e^{-8I} \quad -23 \leq I \leq 22$$

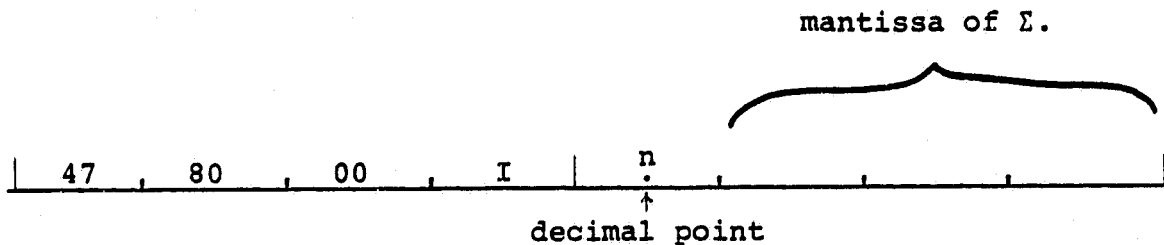
These are arranged in common block XPCOM (which is part of the exponential routine) as follows:

Quantity	Type	Bytes
OUTCNT	I * 4	4
XPXTRA	I * 4	4
XPD	R * 8	8
XPAR	R * 4	1024
XPBIG	R * 4	354

XP procedure:

The argument mantissa is shifted into the lower half of a double word (CARVE), by adding a special constant. If the argument x is $0 \leq x < 16$, the upper word is fixed, and is checked. The upper byte (n) of the mantissa is used as an index to XPAR, and is then replaced with a standard sign-exponent byte, to yield ξ . (The non-normalized character of ξ does not matter because it is immediately added to XPD, which has a larger exponent. This also eliminates the random 4 low-order bits of ξ .) The calculation is then direct. For x outside (0,16), the procedure is the same, except for a range check and an additional factor of XPBIG, derived from an index calculated from the upper half of the double word. If x is out of range (± 352 , corresponding about to $10^{\pm 76}$), the result is set to 0, and overflows are counted.

CARVE structure:



Path Differences in CLASSY

The CLASSY algorithm may take widely different paths to the maximum likelihood solution, depending on very small effects. In some cases, whenever the clusters have not been fully resolved statistically due either to too few iterations or too little data, the two paths may end at slightly different points. Variations causing the algorithm to go on different paths include rounding error (for example use the associative and/or distributive laws) use of new standard function routines (such as the exponential routine), or other minor changes.

The reason CLASSY is so susceptible to small changes in the current version is the use of Monte-Carlo methods in processing the points with respect to classes which give them low probability. Small fluctuations in the calculations may accumulate until the threshold for the Monte-Carlo process is exceeded in one version and not in another. Since the random number in generator is sequential, this offsets the usage of all the succeeding random numbers, giving completely different Monte-Carlo choice for each point.

This, in turn, leads to moderately large fluctuations in the numerical values used, which ultimately changes the point at which the SPLIT or other threshold is passed. Since one cluster being SPLIT and separated can make a major difference in the processing of other clusters, the path taken by CLASSY rapidly becomes quite different.

In summary, minor changes in CLASSY can make large differences in the path the algorithm takes. This is mediated by the amplification caused by various thresholds, in particular those used in the Monte-Carlo subsystem. All the paths should, however, converge to the same point, except when insufficient iterations or data are used.

Appendix 1: Program listings for vector routines

This appendix contains listings of all the routines listed in Table 2. They are in the order: old Fortran, new Fortran, and assembly language. Preceding the routine listings is a listing of the EXEC routines for converting one set to the other.

ORIGINAL PAGE IS
OF POOR QUALITY

SUBROUTINE CORECT (REL,PV,P,S)	COR00010
COMMON /MISC/ HQ,MM,LR,LV,NINCL,MYAR,WTINIT,KROOT,EPS,DELT,	COR00020
1 AHQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,ANM,SBLTH,	COR00030
2 INDVUL,WFAC,NPTSO,PQRATH,SPHVTH,DWFAC,GRACIN,ANOFAC,	COR00040
3 ANOMIN,ANOMAX,ANORAT,VOLLIN,BIAS,PJOIN,VRJOIN,WSIN,WDELSM,	COR00050
4 BETTER,NODE,CORLEN,SPCOR	COR00060
REAL REL(30), PV(30), S(30)	COR00070
DO 10 I = 1,HQ	COR00080
REL(I) = PV(I) - S(I) / P	COR00090
II = I	COR00100
C WRITE (6,9999) II,REL(I),PV(I),S(I),P	COR00110
9999 FORMAT ('CORECT I,REL,PV,S,P',I4,4(E10.4,2X))	COR00120
10 CONTINUE	COR00130
RETURN	COR00140
END	COR00150

ORIGINAL PAGE IS
OF POOR QUALITY

	SUBROUTINE CORECT (REL,PV,P,S)	COR00010
	COMMON /HISC/ HQ,MM,LR,LV,NINCL,MAXAR,WTINIT,KROOT,EPS,DELT,	COR00020
	1 ANQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,AMN,SBLTH,	COR00030
	2 INDVUL,WFAC,NPTSO,PQRATH,SPHVTH,DFAC,GRACIN,AMOFAC,	COR00040
	3 ANOMIN,ANOMAX,AMORAT,VOLLIN,BIAS,PJOIN,VRJOIN,WSIN,WDELSH,	COR00050
	4 BETTER,MODE,CORLEN,SPCOR	COR00060
	REAL REL(30), PV(30), S(30)	COR00070
C		COR00080
	POV=1./P	COR00090
	DO 10 I = 1,HQ	COR00100
	REL(I) = PV(I) - S(I)*POV	COR00110
C	WRITE (6,9999) I,REL(I),PV(I),S(I),P	COR00120
9999	FORMAT ('CORECT I,REL,PV,S,P',I4,4(E10.4,2X))	COR00130
10	CONTINUE	COR00140
	RETURN	COR00150
	END	COR00160

CORECTA

* THIS ROUTINE WAS WRITTEN BY ELOGIC, INC., NOV, 1979, TO REPLACE THE	COR00010
* CORRESPONDING FORTRAN ROUTINE IN THE CLASSY SYSTEM. FOR ADDITIONAL	COR00020
* INFORMATION CONTACT ELOGIC, INC.	COR00030
*	COR00040
* SUBROUTINE CORECT(REL,PV,P,S): REL(I)=PV(I)-S(I)/P	COR00050
CORECT CSECT	COR00060
USING CORECT,15	COR00070
STM 2,9,SAVE	COR00080
LM 1,4,0(1)	COR00090
LA 8,4(0,0)	COR00100
EXTRN MISC	COR00110
L 7,=A(MISC)	COR00120
USING MISC,7	COR00130
L 9,HQ	COR00140
SLA 9,2	COR00150
SR 9,8	COR00160
	COR00170
	COR00180
LE 6,=E'1.0'	COR00190
DE 6,0(3)	COR00200
	COR00210
SR 5,5	COR00220
IC LE 2,0(4,5)	COR00230
MER 2,6	COR00240
LE 0,0(2,5)	COR00250
SER 0,2	COR00260
STE 0,0(1,5)	COR00270
BXLE 5,8,IC	COR00280
	COR00290
LM 2,9,SAVE	COR00300
BR 14	COR00310
SAVE DS 9F	COR00320
* COMMON /MISC/	COR00330
MISC DSECT	COR00340
HQ DS F	COR00350
END	

```

C      FUNCTION DOTSQ(V,AMET)                                DOT00010
C      CALCULATES THE INNER PRODUCT V.V RELATIVE TO THE METRIC AMET DOT00020
C      DOT00030
      DIMENSION LIST(38), XTHP( 8), YTHP( 7), WADJ(24), W(26),DOT00040
1 INDEX(37), LSUBS(40),LSUPER(39), IDADJ(23), NSYMB(36), OW(25),DOT00050
2 PCUM(33),PRIRCH(32), CIN(22), CTOT(21), PROP(29), SPFAC(15),DOT00060
3 OPROP(28), VOLIN(18), VOLRT(17), DCON(16), PORAT(14), ODEN(19),DOT00070
4 DISS(35), PPASS(34), PST(30), OCIN(20), PCOND(31),OPRIOR(27),DOT00080
5 PAVE(13), PILE(12) DOT00090
      DIMENSION VRIN(475),GEN(999),GREF(999),ALINK(99) DOT00100
      EQUIVALENCE (LINK(41),ALINK(41),GREF(8), GEN(7),VRIN(7)) DOT00110
      EQUIVALENCE ( LINK(41), LSUBS(40),LSUPER(39), LIST(38), DOT00120
1 INDEX(37), NSYMB(36), DISS(35), PPASS(34), PCUM(33), DOT00130
2 PRIRCH(32), PCOND(31), PST(30), PROP(29), OPROP(28), DOT00140
3 OPRIOR(27), W(26), OW(25), WADJ(24), IDADJ(23), DOT00150
4 CIN(22), CTOT(21), OCIN(20), ODEN(19), VOLIN(18), DOT00160
5 VOLRT(17), DCON(16), SPFAC(15), PORAT(14), PAVE(13), DOT00170
6 PILE(12), XTHP( 8), YTHP( 7)) DOT00180
      COMMON/CLUS/ JUNK(12),NARL,NTOP,NTBSZH,NWANT,LINK(14000) DOT00190
      DIMENSION MXAR(31),LR(3),LV(3) DOT00200
      EQUIVALENCE (LR(1),LVRIN),(LR(2),LKURT), DOT00210
1 (LR(3),LOVAR),(LV(1),LSUM),(LV(2),LSKEW),(LV(3),LOSUM) DOT00220
C      DOT00230
      COMMON /MISC/ HQ,MH,LR,LV,NINCLS,MXAR,WTINIT,KROOT,EPS,DELT, DOT00240
1 AHQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,AMH,SBLTH, DOT00250
2 INDVVL,WFAC,NPTSO,PQRATH,SPHVTH,DWFAC,GRACFM,ANOFAC, DOT00260
3 ANOMIN,ANOMAX,ANORAT,VOLLIN,BIAS,PJOIN,VRJOIN,WSIM,WDELSH, DOT00270
4 BETTER, MODE DOT00280
C      ( END OF STANDARD BLOCK. ) DOT00290
C      DOT00300
      DIMENSION NTB(32) DOT00310
C      DOT00320
      COMMON /STPAR/WAIT,CONLV,SKBND,SKCHI,TRBND,TRCHI,URKBND,URKCHI, DOT00330
1 PACCEL(2),MACCEL(2),VACCEL(2) DOT00340
      REAL V(30),AMET(475) DOT00350
      REAL*8 DDOTSQ,DGDOT DOT00360
      DDOTSQ=0. DOT00370
      DGDOT=V(1)*V(1)*AMET(1) DOT00380
      DO 10 I=2,MQ DOT00390
      MX=MXAR(I) DOT00400
      7 DO 8 J=2,I DOT00410
      8 DDOTSQ=DDOTSQ+V(I)*V(J-1)*AMET(MX+J-1) DOT00420
      10 DGDOT=DGDOT+V(I)*V(I)*AMET(MX+I) DOT00430
C      THE DIAGONALS ARE HANDLED SEPARATELY BECAUSE EACH OFF- DOT00440
C      DIAGONAL APPEARS TWICE, AND SO MUST BE DOUBLED. DOT00450
      DDOTSQ=DDOTSQ+DDOTSQ+DGDOT DOT00460
      DOTSQ = DDOTSQ DOT00470
      RETURN DOT00480
      END DOT00490

```

```

C      FUNCTION DOTSQ(V,AMET)                                DOT00010
C      CALCULATES THE INNER PRODUCT V.V RELATIVE TO THE METRIC AMET DOT00020
C      DOT00030
      DIMENSION LIST(30), XTMP( 8), YTMP( 7), WADJ(24), W(26), DOT00040
1 INDEX(37), LSUBS(40),LSUPER(39), IDADJ(23), NSYMB(36), OW(25), DOT00050
2 PCUM(33),PRIRCH(32), CIN(22), CTOT(21), PROP(29), SPFAC(15), DOT00060
3 OPROP(28), VOLIN(18), VOLRT(17), DCON(16), PORAT(14), ODEN(19), DOT00070
4 DISS(35), PPASS(34), PST(30), OCIN(20), PCOND(31),OPRIOR(27), DOT00080
5 PAVE(13), PILE(12) DOT00090
      DIMENSION VRIN(475),GEN(999),GREF(999),ALINK(99) DOT00100
      EQUIVALENCE (LINK(41),ALINK(41),GREF(8), GEN(7),VRIN(7)) DOT00110
      EQUIVALENCE ( LINK(41), LSUBS(40),LSUPER(39), LIST(38), DOT00120
1 INDEX(37), NSYMB(36), DISS(35), PPASS(34), PCUM(33), DOT00130
2 PRIRCH(32), PCOND(31), PST(30), PROP(29), OPROP(28), DOT00140
3 OPRIOR(27), W(26), OW(25), WADJ(24), IDADJ(23), DOT00150
4 CIN(22), CTOT(21), OCIN(20), ODEN(19), VOLIN(18), DOT00160
5 VOLRT(17), DCON(16), SPFAC(15), PORAT(14), PAVE(13), DOT00170
6 PILE(12), XTMP( 8), YTMP( 7)) DOT00180
      COMMON/CLUS/ JUNK(12),NARL,NTOP,NTBSZH,NWANT,LINK(14000) DOT00190
      DIMENSION MXAR(31),LR(3),LV(3) DOT00200
      EQUIVALENCE (LR(1),LVRIN),(LR(2),LKURT), DOT00210
1 (LR(3),LOVAR),(LV(1),LSUM),(LV(2),LSKEW),(LV(3),LOSUM) DOT00220
C      DOT00230
      COMMON /MISC/ MQ,MH,LR,LV,NINCLS,MXAR,WTINIT,KROOT,EPS,DELT, DOT00240
1 AMQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,AMH,SDLTH, DOT00250
2 INDVUL,WFAC,NPTSQ,PORATH,SPHVTH,DUFAC,GRACTH,AMOFAC, DOT00260
3 AMOHIN,AMOMAX,AMORAT,VOLLIN,BIAS,PJOIN,VRJOIN,WSIH,WDELSH, DOT00270
4 BETTER, MODE DOT00280
C      ( END OF STANDARD BLOCK. ) DOT00290
C      DOT00300
      DIMENSION NTB(32) DOT00310
C      DOT00320
      COMMON /STPAR/ WAIT,CONLV,SKBND,SKCHI,TRBND,TRCHI,URKBND,URKCHI, DOT00330
1 PACCEL(2),MACCEL(2),VACCEL(2) DOT00340
      REAL V(30),AMET(475) DOT00350
      DOTSQ=V(1)*V(1)*AMET(1) DOT00360
      DO 10 I=2,MQ DOT00370
      SQ=0. DOT00380
      MX=MXAR(I) DOT00390
7 DO 8 J=2,I DOT00400
8 SQ=SQ+V(J-1)*AMET(MX+J-1) DOT00410
10 DOTSQ=DOTSQ+V(I)*((V(I)*AMET(MX+I)+SQ)+SQ) DOT00420
C      THE DIAGONALS ARE HANDLED SEPARATELY BECAUSE EACH OFF- DOT00430
C      DIAGONAL APPEARS TWICE, AND SO MUST BE DOUBLED. DOT00440
      RETURN DOT00450
      END DOT00460

```

```

* THIS ROUTINE WAS WRITTEN BY ELOGIC, INC., NOV, 1979, TO REPLACE THE
* CORRESPONDING FORTRAN ROUTINE IN THE CLASSY SYSTEM. FOR ADDITIONAL
* INFORMATION CONTACT ELOGIC, INC.
*
*FUNCTION DOTSQ(V,AMET): CALCULATE THE INNER PRODUCT
*      GBLC      &P      ADD - SUBTRACT PRECISION
&P      SETC      'ER'      ER SINGLE PRECISION; DR- DOUBLE PREC
DOTSQ      CSECT
          USING DOTSQ,15
          STM      2,12,SAVE
          LM        1,2,0(1)      ARGS ADDR
          LA        4,4(0,0)      I INCREMENT IS 1
          EXTRN     MISC
          L         11,=A(MISC)
          USING     MISC0,11
          L         5,MQ          I COMPARAND IS MQ
          SLA       5,2          MULTIPLY BY 4
*
          LR        6,4          J INCREMENT IS 1
          LA        8,8(0,0)
          SR        1,8          INDEX OF V STARTS FROM 1
          SR        2,8          INDEX OF AMET STARTS FROM 1
*
          LE        0,4(1,4)      V(1)
          MER       0,0          V(1)*V(1)
          ME        0,4(2,4)      *AMET(1)
          S&P       2,2          0.
*
          LR        7,8          DO LOOP INDEX I STARTS FROM 2
I1      L&P        4,2          SQ=0.
          L         12,MXAR-4(7)  MX=MXAR(I)
          SLA       12,2          MULTIPLY BY 4
          AR        12,2          ADD ADDR AMET
*
          LR        9,8          DO LOOP INDEX J STARTS FROM 2
I2      LE        6,0(1,9)      V(J-1)
          ME        6,0(12,9)    V(J-1)*AMET(MX+J-1)
          A&P       4,6          SQ
          BXLE      9,6,I2

```

ORIGINAL PAGE IS
OF POOR QUALITY

*	LE	6,4(1,7)	V(I)	DOT00410
	HE	6,4(12,7)	V(I)*AMET(MX+I)	DOT00420
	A&P	6,4	+SQ	DOT00430
	A&P	6,4	+SQ	DOT00440
	HE	6,4(1,7)	*V(I)	DOT00450
	A&P	0,6	DOTSQ	DOT00460
	BXLE	7,4,I1		DOT00470
*				DOT00480
	LH	2,12,SAVE		DOT00490
	BR	14		DOT00500
*	SAVE	DS 12F		DOT00510
*				DOT00520
*				DOT00530
*	COMMON	/MISC/		DOT00540
	MISC	DSECT		DOT00550
	HQ	DS F		DOT00560
	HM	DS F		DOT00570
	LR	DS 3F		DOT00580
	LV	DS 3F		DOT00590
	NINCLS	DS F		DOT00600
	MXAR	DS 31F		DOT00610
	END			DOT00620
				DOT00630

ORIGINAL PAGE IS
OF POOR QUALITY

	SUBROUTINE MPVS(AM,C,V)	
C	SETS AM=AM+V*V*C (TENSOR PRODUCT)	MPV00010
C		MPV00020
	COMMON /MISC/ HQ,MH,LR,LV,NINCL,MXAR,UTINIT,KROOT,EPS,DELT,	MPV00030
1	AMQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,AMM,SBLTH,	MPV00040
2	INDXVL,UFAC,NPTSO,PQRATH,SPMUTH,DWFAC,GRACM,AMOFAC,	MPV00050
3	AMOMIN,AMOMAX,AMORAT,VOLLIN,BIAS,PJOIN,VRJOIN,USIM,WDELSM,	MPV00060
4	BETTER,MODE,CORLEN,SPCOR	MPV00070
		MPV00080
C	REAL AM(475),V(30)	MPV00090
	LOC=0	MPV00100
	DO 10 I=1,MQ	MPV00110
	DO 10 J=1,I	MPV00120
	LOC=LOC+1	MPV00130
10	AM(LOC)=AM(LOC)+V(I)*V(J)*C	MPV00140
	RETURN	MPV00150
	END	MPV00160
		MPV00170

ORIGINAL PAGE IS
OF POOR QUALITY

	SUBROUTINE MPVS(AH,C,V)	MPV00010
C	SETS AH=AH+V*V (TENSOR PRODUCT)	MPV00020
C		MPV00030
	COMMON /MISC/ HQ,MM,LR,LV,NINCL,MAXAR,WTINIT,KROOT,EPS,DELT,	MPV00040
	1 AMQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINT,SEPT,VFAC,AMH,SBLTH,	MPV00050
	2 INDVVL,WFAC,NPTSO,PORATH,SPMVTH,DWFAC,GRACHT,AMOFAC,	MPV00060
	3 AMOMIN,AMOMAX,AMORAT,VOLLIN,BIAS,PJOIN,VRJOIN,WSIN,WDELSN,	MPV00070
	4 BETTER,MODE	MPV00080
C		MPV00090
	REAL AH(475),V(30)	MPV00100
	LOC=0	MPV00110
	DO 10 I=1,HQ	MPV00120
	CV=C*V(I)	MPV00130
	DO 10 J=1,I	MPV00140
	LOC=LOC+1	MPV00150
	10 AH(LOC)=AH(LOC)+V(J)*CV	MPV00160
	RETURN	MPV00170
	END	MPV00180

ORIGINAL PAGE IS
OF POOR QUALITY

page 21

MPVSA

```

* THIS ROUTINE WAS WRITTEN BY ELOGIC, INC., NOV, 1979, TO REPLACE THE
* CORRESPONDING FORTRAN ROUTINE IN THE CLASSY SYSTEM. FOR ADDITIONAL
* INFORMATION CONTACT ELOGIC, INC.
*
* SUBROUTINE MPVS(AH,C,V): SETS AH=AH+V*V*C
MPVS      CSECT
          USING MPVS,15
          STM 2,12,SAVE
          LM  1,3,0(1)
          LA  4,4(0,0)
          EXTRN MISC
          L   12,=A(MISC)
          USING MISC0,12
          L   5,MQ
          SLA 5,2
          SR  5,4
          LR  8,4
          SR  7,7
          SR  9,9
          LE  2,0(2)
          LER 6,2
          ME  6,0(3,9)
          SR  11,11
          LER 0,6
          ME  0,0(3,11)
          AE  0,0(1,7)
          STE 0,0(1,7)
          AR  7,4
          BXLE 11,8,12
          BXLE 9,4,11
          LM  2,12,SAVE
          BR  14
          SAVE DS 12F
* COMMON /MISC/
MISC0 DSECT
MQ DS F
END
          ARGV ADDR
          I INCREMENT IS 1
          COMPARAND IS MQ
          MULTIPLY BY 4
          SUBTRACT 4
          J INCREMENT IS 1
          LOC
          DO LOOP INDEX I
          C
          C*V(I)
          DO LOOP INDEX J
          C*V(I)
          C*V(I)*V(J)
          ADD AH(LOC)
          STORE AH(LOC)
          ADD 1 TO LOC

```

NPV00010
 NPV00020
 NPV00030
 NPV00040
 NPV00050
 NPV00060
 NPV00070
 NPV00080
 NPV00090
 NPV00100
 NPV00110
 NPV00120
 NPV00130
 NPV00140
 NPV00150
 NPV00160
 NPV00170
 NPV00180
 NPV00190
 NPV00200
 NPV00210
 NPV00220
 NPV00230
 NPV00240
 NPV00250
 NPV00260
 NPV00270
 NPV00280
 NPV00290
 NPV00300
 NPV00310
 NPV00320
 NPV00330
 NPV00340
 NPV00350
 NPV00360
 NPV00370
 NPV00380
 NPV00390
 NPV00400
 NPV00410
 NPV00420

C	SUBROUTINE VMTV(VA,AMET,VB)	VMT00010
	SETS VA=AMET*VB	VMT00020
	COMMON /HISC/ HQ,MM,LR,LV,NINCLS,MXAR,WTINIT,KROOT,EPS,DELT,	VMT00030
1	AMQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,AMH,SBLTH,	VMT00040
2	INDXVL,WFAC,NPTSD,PQRATH,SPHVTH,DFAC,GRACTH,AMOFAC,	VMT00050
3	AMOHIN,AMOHAX,AMORAT,VOLLIN,BIAS,PJOIN,VRJOIN,USTH,USLSH,	VMT00060
4	BETTER,MODE,CORLEN,SPCOR	VMT00070
	REAL VA(30),VB(30),AMET(475)	VMT00080
	LOCA=0	VMT00090
	DO 20 I=1,HQ	VMT00100
	SUM=0.	VMT00110
	DO 10 J=1,I	VMT00120
	LOCA=LOCA+1	VMT00130
10	SUM=SUM+ AMET(LOCA)*VB(J)	VMT00140
	IF(I.EQ.HQ) GO TO 20	VMT00150
	JS=I+1	VMT00160
	LOCB=LOCA+I	VMT00170
	DO 11 J=JS,HQ	VMT00180
	SUM=SUM+AMET(LOCB)*VB(J)	VMT00190
11	LOCB=LOCB+J	VMT00200
20	VA(I)=SUM	VMT00210
	RETURN	VMT00220
	END	VMT00230

VMTVA

```

* THIS ROUTINE WAS WRITTEN BY ELOGIC, INC., NOV, 1979, TO REPLACE VMT00010
* CORRESPONDING FORTRAN ROUTINE IN THE CLASSY SYSTEM. FOR ADDITIONAL VMT00020
* INFORMATION CONTACT ELOGIC, INC. VMT00030
* VMT00040
* SUBROUTINE VMTV(VA, AMET, VB): SETS VA=AMET*VB VMT00050
VMTV CSECT VMT00060
      USING VMTV, 15 VMT00070
      STM 2, 12, SAVE VMT00080
      LM 1, 3, 0(1) VMT00090
      LA 4, 4(0, 0) VMT00100
      EXTRN MISC VMT00110
      L 7, =A(MISC) VMT00120
      USING MISC, 7 VMT00130
      L 5, HQ VMT00140
      SLA 5, 2 VMT00150
      DROP 7 VMT00160
* VMT00170
      LR 8, 4 VMT00180
      SR 1, 4 VMT00190
      SR 2, 4 VMT00200
      SR 3, 4 VMT00210
      SR 7, 7 VMT00220
* VMT00230
      LR 9, 4 VMT00240
      LE 6, =E'0.0' VMT00250
* VMT00260
      LR 11, 4 VMT00270
      AR 7, 4 VMT00280
      LE 0, 0(2, 7) VMT00290
      ME 0, 0(3, 11) VMT00300
      AER 6, 0 VMT00310
      BXLE 11, 8, 12 VMT00320
* VMT00330
      CR 9, 5 VMT00340
      BZ A12 VMT00350
      LR 11, 9 VMT00360
      AR 11, 4 VMT00370
      LR 12, 7 VMT00380
      AR 12, 9 VMT00390
      LE 0, 0(2, 12) VMT00400
      ME 0, 0(3, 11) VMT00410
      AER 6, 0 VMT00420
      AR 12, 11 VMT00430
      BXLE 11, 4, 13 VMT00440
* VMT00450
      STE 6, 0(1, 9) VMT00460
      BXLE 9, 4, 11 VMT00470
* VMT00480
      LM 2, 12, SAVE VMT00490
      BR 14 VMT00500
      DS 12F VMT00510
* COMMON /MISC/ VMT00520
      MISC DSECT VMT00530
      HQ DS F VMT00540
      END VMT00550

```

	SUBROUTINE VPV(VA,FAC,VB)	VPV00010
C	SETS VA=VA+FAC*VB	VPV00020
C		VPV00030
	COMMON /MISC/ HQ,MH,LR,LV,NINCLS,MXAR,WTINIT,KROOT,EPS,DELT,	VPV00040
1	AMQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,AMH,SBLTH,	VPV00050
2	INDXVL,WFAC,NPTSO,PQRATH,SPHVTH,DWFAC,GRACIN,AMOFAC,	VPV00060
3	AMOMIN,AMOMAX,AMORAT,VOLLIN,BIAS,PJOIN,VRJOIN,WSIN,WDELSM,	VPV00070
4	BETTER,MODE,CORLEN,SPCOR	VPV00080
C		VPV00090
	REAL VA(30),VB(30)	VPV00100
	DO 10 I=1,HQ	VPV00110
10	VA(I)=VA(I)+VB(I)*FAC	VPV00120
	RETURN	VPV00130
	END	VPV00140

ORIGINAL PAGE IS
OF POOR QUALITY

```

* THIS ROUTINE WAS WRITTEN BY ELOGIC, INC., NOV, 1979, TO REPLACE THE
* CORRESPONDING FORTRAN ROUTINE IN THE CLASSY SYSTEM. FOR ADDITIONAL
* INFORMATION CONTACT ELOGIC, INC.
*
* SUBROUTINE VPV(VA,FAC,VB): VA(I)=VA(I)+FAC*VB(I)
VPV      CSECT
        USING VPV,15
        STM 2,7,SAVE
        LM  1,3,0(1)      ARG5 ADDR
        LE  2,0(2)      ARG2
        LA  4,4(0,0)      INCREMENT IS 1
        EXTRN MISC
        L   7,=A(MISC)
        USING MISC,7
        L   5,MQ          COMPARAND IS MQ
        SLA 5,2          MULTIPLY BY 4
        SR  5,4          SUBTRACT 4
*
        SR  2,2          DO LOOP INDEX I STARTS FROM 1
IC       LER 0,2          FAC
        HE  0,0(3,2)      FAC*VB(I)
        AE  0,0(1,2)      VA(I)
        STE 0,0(1,2)      STORE VA(I)
        BXLE 2,4,IC      BRANCH LOCATION
*
        LM  2,7,SAVE
        BR  14
SAVE     DS  7F
* COMMON /MISC/
MISC     DSECT
MQ       DS  F
        END

```

VPV00010
VPV00020
VPV00030
VPV00040
VPV00050
VPV00060
VPV00070
VPV00080
VPV00090
VPV00100
VPV00110
VPV00120
VPV00130
VPV00140
VPV00150
VPV00160
VPV00170
VPV00180
VPV00190
VPV00200
VPV00210
VPV00220
VPV00230
VPV00240
VPV00250
VPV00260
VPV00270
VPV00280
VPV00290
VPV00300
VPV00310
VPV00320

Appendix 2: Program listings for the fast exponential system.

This appendix contains the routines used by the fast exponential system. They include the routine XP, the table generator XPREP, the modified version of CLINIT, showing where XPREP is called; and the modified version of STATIS, showing the calls to XP. (The version of STATIS displayed also contains modifications for the additional statistics collection, which will be the subject of an additional report.) The modification to CLINIT consists only of the one line calling XPREP, which may be executed even if XP is not used. Elogic recommends that the version containing the call to XPREP be used at all times.

```

* THIS ROUTINE WAS WRITTEN BY ELOGIC, INC., NOV, 1979, TO REPLACE THE
* CORRESPONDING FORTRAN ROUTINE IN THE CLASSY SYSTEM. FOR ADDITIONAL
* INFORMATION CONTACT ELOGIC, INC.
*
* EXPONENTIAL ROUTINE, EXP(-X/2)
XP      CSECT
      USING      *,15
      L          1,0(1)      ARG ADDR
      LE         0,0(1)      ARG
      AD         0,SHIFWRD    EXTRACT MANTISSA
      STD        0,CARVE
      L          1,CARVE      GET UPPERWORD
      S          1,UPCARVE    CHK ARG BETWEEN
      BNZ        NSEXP        0 AND 16
XPSEQ   IC        1,CARVE+4    GET UPPER BYTE
      MVI        CARVE+4,X'3F' PUT IN EXPONENT
      LE         0,CARVE+4    LOAD XI
      AE         0,XPD        CALC EXPRESSION
      ALR        1,1          INDEX
      ^LR        1,1
      ME         0,XPAR(1)    EXPONENTIAL TABLE
      BR         14          (BYE)
*
NSEXP   C          1,MINCARV    CHK UNDERFLOW
      BM         ZEROIT
      C          1,MAXCARV    CHK OVERFLOW
      BP         OUTRANGE
      SR         1,1
      IC        1,CARVE+4      (DUP XPSEQ)
      MVI        CARVE+4,X'3F'
      LE         0,CARVE+4
      AE         0,XPD
      ALR        1,1
      ALR        1,1
      ME         0,XPAR(1)
      L          1,CARVE      GET BIG PART
      ALR        1,1          INDEX -USES SPEC
      ALR        1,1          PROPERTIES OF SHIFWRD
      ME         0,XPBIG(1)    BIG EXPONENTIAL TABLE
      BR         14          (BYE)
*
ZEROIT  LE         0,=F'0'      UNDERFLOW -
      BR         14          RETURN 0.
*
OUTRANGE L        1,OUTCNT
      A          1,=F'1'
      ST         1,OUTCNT
      B          ZEROIT

```

XP 00010
 XP 00020
 XP 00030
 XP 00040
 XP 00050
 XP 00060
 XP 00070
 XP 00080
 XP 00090
 XP 00100
 XP 00110
 XP 00120
 XP 00130
 XP 00140
 XP 00150
 XP 00160
 XP 00170
 XP 00180
 XP 00190
 XP 00200
 XP 00210
 XP 00220
 XP 00230
 XP 00240
 XP 00250
 XP 00260
 XP 00270
 XP 00280
 XP 00290
 XP 00300
 XP 00310
 XP 00320
 XP 00330
 XP 00340
 XP 00350
 XP 00360
 XP 00370
 XP 00380
 XP 00390
 XP 00400
 XP 00410
 XP 00420
 XP 00430
 XP 00440
 XP 00450
 XP 00460
 XP 00470
 XP 00480

ORIGINAL PAGE IS
OF POOR QUALITY

page 28

*				XP 00490
*				XP 00500
CARVE	DS	D		XP 00510
SHIFWRD	DC	A(X'47800000')	SHIFTS ARGUMENT TO GET	XP 00520
	DC	F'0'	MANTISSA	XP 00530
UPCARVE	EQU	SHIFWRD	ELIMS SHIFWRD	XP 00540
MINCARV	DC	F'-21'		XP 00550
MAXCARV	DC	F'21'		XP 00560
*				XP 00570
XPCOM	DS	0D		XP 00580
	ENTRY	XPCOM		XP 00590
OUTCNT	DC	F'0'		XP 00600
XPXTRA	DC	F'0'		XP 00610
XPD	DS	D		XP 00620
XPAR	DS	256F		XP 00630
	DS	21F	XPBIG NEG ARG	XP 00640
XPBIG	DS	22F		XP 00650
	END			XP 00660

	SUBROUTINE XPREP	XPR00010
	COMMON /XPCOM/ OUTCNT,XPXTRA,XPD,XPARO(256),XPBIGO(43)	XPR00020
	REAL*8 XPD,ALPHA,C2,D1	XPR00030
C		XPR00040
	BETA=1./16.	XPR00050
	ALPHA=BETA/4.	XPR00060
	C2=2.*ALPHA	XPR00070
	D1=3.*(DSINH(ALPHA)-ALPHA*DCOSH(ALPHA))/(C2*ALPHA*ALPHA)	XPR00080
	XPD=DSINH(ALPHA)/(ALPHA*D1) - C2	XPR00090
	DO 10 N=1,256	XPR00100
10	XPARO(N)=D1*DEXP(-C2*(N-1)-ALPHA)	XPR00110
	DO 20 I=1,43	XPR00120
20	XPBIGO(I)=DEXP(-8.D0*(I-22))	XPR00130
	RETURN	XPR00140
	END	XPR00150

```

SUBROUTINE CLINIT(KROT)                                CLI00010
C THIS ROUTINE CONTAINS THE VARIOUS STATEMENTS NECESSARY TO  CLI00020
C INITIALIZE THE CLUSTERING ALGORITHM.                  CLI00030
C                                                         CLI00040
  DIMENSION LIST(38), XTMP( 8), YTMP( 7), WADJ(24),      W(26), CLI00050
1 INDEX(37), LSUBS(40),LSUPER(39), IDADJ(23), NSYMB(36),   OW(25), CLI00060
2 PCUM(33),PRIRCH(32), CIN(22), CTOT(21), PROP(29), SPFAC(15), CLI00070
3 OPROP(28), VOLIN(18), VOLRT(17), DCON(16), PQRAT(14), ODEN(19), CLI00080
4 DISS(35), PPASS(34), PST(30), OCIN(20), PCOND(31),OPRIOR(27), CLI00090
5 PAVE(13), PILE(12)                                     CLI00100
  DIMENSION VRIN(475),GEN(999),GREF(999),ALINK(99)        CLI00110
  EQUIVALENCE (LINK(41),ALINK(41),GREF(8), GEN(7),VRIN(7)) CLI00120
  EQUIVALENCE ( LINK(41), LSUBS(40),LSUPER(39), LIST(38), CLI00130
1 INDEX(37), NSYMB(36), DISS(35), PPASS(34), PCUM(33),    CLI00140
2 PRIRCH(32), PCOND(31), PST(30), PROP(29), OPROP(28),    CLI00150
3 OPRIOR(27), W(26), OW(25), WADJ(24), IDADJ(23),        CLI00160
4 CIN(22), CTOT(21), OCIN(20), ODEN(19), VOLIN(18),      CLI00170
5 VOLRT(17), DCON(16), SPFAC(15), PQRAT(14), PAVE(13),   CLI00180
6 PILE(12), XTMP( 8), YTMP( 7))                         CLI00190
  COMMON/CLUS/ JUNK(12),NARL,NTOP,NTBSZH,NWANT,LINK(14000) CLI00200
  DIMENSION MXAR(31),LR(3),LV(3)                         CLI00210
  EQUIVALENCE (LR(1),LVRIN),(LR(2),LKURT),               CLI00220
1 (LR(3),LOVAR),(LV(1),LSUM),(LV(2),LSKEW),(LV(3),LOSUM) CLI00230
C                                                         CLI00240
  COMMON /MISC/ HQ,HM,LR,LV,NINCLS,MXAR,WTINIT,KROOT,EPS,DELT, CLI00250
1 AMQ,ODCON,XOVFLO,XUNFLO,WADJIN,ELINTH,SEPTH,VFAC,AMM,SBLTH, CLI00260
2 INDXVL,WFAC,NPTSD,PQRATH,SPMVTH,DWFAC,GRACHT,AMOFAC,    CLI00270
3 AMOMIN,AMOMAX,AMORAT,VOLLIN,BIAS,PJOIN,VRJOIN,WSIN,WDELSH, CLI00280
4 BETTER, MODE                                           CLI00290
C ( END OF STANDARD BLOCK. )                             CLI00300
C                                                         CLI00310
  DOUBLE PRECISION XTEMP,YTEMP,ZTEMP,DURK,DURKD          CLI00320
C                                                         CLI00330
  COMMON /STPAR/WAIT,CONLV,SKBND,SKCHI,TRBND,TRCHI,URKBND,URKCHI, CLI00340
1 PACCEL(2),MACCEL(2),VACCEL(2)                         CLI00350
C                                                         CLI00360
  COMMON/CLUSTR/ IBEGIN,TOTWRD,CLSNAH,IPT,NOFLD, SYM(61) , CLI00370
1 LNCAT, PRNT(4) , KLBC , PRTHE, PROUT, TOTPIX,          CLI00380
2 SCRAM1,BUFFIX,BUFTOT,NBUFS,NDUMP,LBUFD                CLI00390
3, MAXBF, AREA, NWDS, NWDRS, NPTS, LBUF, IQ1,NOCYCL, NCL  CLI00400
C                                                         CLI00410
  INTEGER TOTWRD,SYM,PRNT,PRTHE,PROUT,TOTPIX,SCRAM1,BUFFIX,BUFTOT CLI00420
1 ,CLSNAH                                                CLI00430
C                                                         CLI00440
  COMMON /MXLL/ MXLLWT, MXLLFN, RELPRP(200)              CLI00450
C                                                         CLI00460
  COMMON /INITL/WTNEW,DEVINI,CHANIN                     CLI00470
  CHIVAL(DF)=DF*(1.-.222/DF+CONLV*SQRT(.222/DF))*3      CLI00480

```

ORIGINAL PAGE IS
OF POOR QUALITY

C INITIALIZE ADDRESS	CLI00490
C	CLI00500
CALL DATFIX	CLI00510
C	CLI00520
AMQ=MQ	CLI00530
MQS=MQ*MQ	CLI00540
C	CLI00550
C DEFINE VALUE OF SEPTH IN TERMS OF CHI SQUARE VALUE	CLI00560
DFT = AMQ + 1	CLI00570
SEPTH = (CHIVAL(DFT))/2	CLI00580
C	CLI00590
C WE FIRST SET UP VARIOUS INDEX ARRAYS FOR A PARTICULAR	CLI00600
C NUMBER OF CHANNELS MQ.	CLI00610
C SET UP THE TRIANGULAR POSITION ARRAY MXAR.	CLI00620
MM=0	CLI00630
DO 10 I=1,31	CLI00640
MXAR(I)=MM	CLI00650
10 MM=MM+1	CLI00660
MM=MXAR(MQ+1)	CLI00670
AMM=MM	CLI00680
C SET UP TABLES FOR THE XP FUNCTION	CLI00690
CALL XPREP	CLI00700
C NOW WE SET UP THE ORIGIN VECTORS, LR AND LV, OF THE VARIOUS ARRAYS	CLI00710
C AND VECTORS IN A CLUSTER NODE.	CLI00720
NINCLS=1	CLI00730
C ***** THIS CONSTANT MUST BE SET TO THE NUMBER OF ARRAYS *****	CLI00740
DO 21 I=1,3	CLI00750
LR(I)=NINCLS	CLI00760
21 NINCLS=NINCLS+MM	CLI00770
DO 22 I=1,3	CLI00780
LV(I)=NINCLS	CLI00790
22 NINCLS=NINCLS+MQ	CLI00800
NSCALS = 35	CLI00810
NINCLS=NINCLS+NSCALS-1	CLI00820
C WE MUST ALSO SET UP SOME THRESHOLDS FOR USE BY THE STATISTICAL	CLI00830
C SYSTEM.	CLI00840
SKCHI=(AMQ+2.)*(AMQ+4.)*CHIVAL(AMQ)	CLI00850
URKCHI=AMQ*(AMQ+4.)*(AMQ+6.)/(AMQ-.999)*CHIVAL(AMM-1.)	CLI00860
TRCHI=CONLV*CONLV*(AMQ*(AMQ+2.)*(AMQ+3.)*8.)	CLI00870
C WE NOW CREATE THE HEAD NODE OF THE CLUSTER TREE. THIS IS NOT	CLI00880
C AN ACTUAL CLUSTER, AND DOES NOT HAVE STORAGE FOR ANY	CLI00890
C OF THE STATISTICAL ARRAYS.	CLI00900
NPTSO=0	CLI00910
KROT=HORSTR(NSCALS)	CLI00920
C MAKE FIRST NODE START AT AN ODD NUMBER	CLI00930
IF (MOD(NTOP,2) .NE. 1) NTOP = NTOP + 1	CLI00940
LINK(KROT)=-262139	CLI00950
LSUPER(KROT)=-262142	CLI00960

ORIGINAL PAGE IS
OF POOR QUALITY.

page 32

IDADJ(KROT)=999999	CLIO0970
INDEX(KROT)=-0	CLIO0980
SPFAC(KROT)=99999.	CLIO0990
W(KROT)=WTINIT	CLIO1000
OW(KROT)=W(KROT)	CLIO1010
PQRAT(KROT)=0.	CLIO1020
PROP(KROT)=1.	CLIO1030
OPROP(KROT)=1.	CLIO1040
CIN(KROT)=W(KROT)	CLIO1050
OCIN(KROT)=CIN(KROT)	CLIO1060
CTOT(KROT)=0.	CLIO1070
ODEN(KROT)=W(KROT)	CLIO1080
PRIRCH(KROT)=1.	CLIO1090
C NEXT THE INITIAL NODE IS SET UP, TOGETHER WITH SOME CONTROL THRESHOLD	CLIO1100
ODCON =60.	CLIO1110
PROPI = 1.	CLIO1120
INDXVL = 0	CLIO1130
57 KFIR = NEWCLS(KROT, PROPI, WTINIT)	CLIO1140
LSUBS(KROT) =KFIR	CLIO1150
LINK(KFIR)=0	CLIO1160
DEV2WT =DEVINI*DEVINI*WTINIT	CLIO1170
LA=MORSTR(MQS)	CLIO1180
DO 52 I=1,MQ	CLIO1190
GREF(KFIR+LSUM+I)=WTINIT*CHANIN	CLIO1200
DO 52 J=1,MQ	CLIO1210
IJ=J-1+(I-1)*MQ+LA	CLIO1220
ALINK(IJ) =0.	CLIO1230
IF(I.EQ.J) ALINK(IJ)=DEV2WT	CLIO1240
52 CONTINUE	CLIO1250
C	CLIO1260
CALL ZEROCL(KFIR,ALINK(LA),GREF(KFIR+LSUM+1),GREF(KFIR+LSKEW+1),	CLIO1270
1 GREF(KFIR+LKURT+1), GREF(KFIR+LOSUM+1),GREF(KFIR+LOVAR+1))	CLIO1280
CALL FREE(LA, MQS)	CLIO1290
C SET SWITCHES FOR MAX LIKLIHOOD LABELING	CLIO1300
MXLLWT = 0	CLIO1310
WADJ(KFIR) =WADJIN	CLIO1320
MXLLFN = 23	CLIO1330
PRINT 273,MQ,CONLV,TRCHI,SKCHI,URKCHI	CLIO1340
273 FORMAT ('1 CONFIDENCE LEVELS',I4,' CHANNELS',F8.4,' CHISQUARES',	CLIO1350
1 3F12.2)	CLIO1360
C	CLIO1370
C	CLIO1380
C INITIALIZE ADDRESS :	CLIO1390
C	CLIO1400
CALL STATAD(GEN(LSUM),GEN(LSKEW),GEN(LKURT),	CLIO1410
1 GEN(LOSUM),GEN(LOVAR))	CLIO1420
RETURN	CLIO1430
END	CLIO1440